Numerical treatment of interfaces for second-order wave equations

Mariana Cécere, Florencia Parisi, and Oscar Reula*

FaMAF-Universidad Nacional de Córdoba, IFEG-CONICET, Ciudad Universitaria, 5000, Córdoba, Argentina

December 15, 2011

Abstract

In this article we develop a numerical scheme to deal with interfaces between touching numerical grids when solving the second-order wave equation. In the spirit of the Simultaneous Approximation Term (SAT) schemes introduced in [1], information is passed among grids using the values of the fields only at the contact points between them (actually, in our case, just the values of the field corresponding to the time derivative of the field). The scheme seems to be as accurate as the space and time discretizations used for the corresponding derivatives. The semi-discrete approximation preserves the norm and uses standard finite-difference operators satisfying summation by parts. For the time integrator we use a semi-implicit IMEX Runge-Kutta method. This is crucial for, otherwise, the methods will be impractical given the severe restrictions its stiff parts would put on totally explicit integrators.

1 Introduction

In many instances, modern treatments of physical models based on systems of partial differential equations require the use of several grid patches. This could be because the topology of the underlying space is not trivial and can not be described by a unique chart (as is the case of codes evolving fields on a sphere; for instance, the earth surface) or it could be because the problem is too large for a single computer and so it has to be partitioned to be solved in parallel in clusters of computers. Or it could be because some areas of the integration domain need more resolution than others. Information needs to be passed among these grids in a dynamical and synchronized fashion. So it is important to devise methods that guarantee the stability of the global solution, and to study the minimal amount of information that needs to be transferred at grid interfaces in order to preserve a given accuracy. For some years now there has been available numerical techniques to deal with these interface problems when solving first-order hyperbolic or parabolic equations. Some of them use interpolation between regions of overlap, while others use penalties, which modify the system at boundary grid points by including information from the same space points at other grids [1]. This last method is preferable in many situations, for it has very nice properties. The most interesting of these is the fact that it is constructed so that the resulting semi-discrete system preserves the corresponding continuum energy estimate of the corresponding constant-coefficient linear systems. Thus we can ensure that, at least for linear, constant-coefficient systems, the scheme is stable. Another property that makes these schemes attractive is that the amount of information one has to pass from one grid to the next is minimal. Thus these schemes are optimal for massive parallel computations, where calculations must be split among several CPU/GPU's, and communication among them adds a non-trivial overhead to the computation of the solution. Furthermore, the method is as accurate as the rest of the time and space discretizations allow, while keeping constant the data transferred. This is not the case, for instance, when interpolation is used, for in order to increase the accuracy of that scheme one has to take more points on each of the grids and to use higher-order interpolants.

A drawback of these standard schemes is that, for a second-order hyperbolic equation, one must first rewrite it as a first-order system in order to be able to apply the approach, thus creating many more variables

*cecere@famaf.unc.edu.ar, florparisi@gmail.com, oreula@gmail.com.

and constraints, all of which have to be monitored during the evolution. This not only increases the required machine memory, but also increases the amount of traffic among processors, due to the corresponding increase on information that needs to be passed along the boundaries.

Recently one of us [2] found a way to implement similar techniques for Schrödinger's equation. Here, we show that the same underlying idea can be extended to deal also with the second-order wave equation. In this scheme only one field needs to be passed across the boundaries, namely the time derivative of the field, making this a very efficient and simple algorithm.

In the next section we describe the new numerical scheme, and derive the boundary terms needed to be added to the equations for this method to work. There are two different types of boundary terms. One of them includes only the values of the fields at the same grid: Its presence is needed to cancel, in the energy estimate, the usual boundary term from the elliptic part of the operator. The other term can be regarded as an interaction between the fields at both grids and it is a penalty in the sense that it depends on the difference of fields on both sides of the grids and drives that difference exponentially to zero. One of the boundary terms –the one that has the information from the neighboring grid– is highly stiff, so an implicit method is needed in order not to pay a huge price in the time step. For that we use IMEX Runge-Kutta methods, which solve the problem in a very general way.

In section 3 we present some numerical results using this method. We compare the results of evolving a one-dimensional system on a circle, first using periodic boundary conditions in a single grid (that is, using a homogeneous scheme with centered difference operators), and then using the interface scheme between the first and final point of the grid. We also compare our scheme with the usual SAT scheme, in which the system is treated in its first-order form. In addition, we study also an implementation in a two-dimensional torus made out of a square grid by identifying their boundaries in the usual way. In one direction we use periodic boundary conditions in the standard way, namely using centered finite-difference operators, while in the other we use an interface scheme, either the one proposed here, the SAT one (using the first-order version of the system), or the periodic one. We first evolve very smooth initial data in order to study convergence and stability, and then quite rough data, namely, a wave packet with a definite trajectory, to study the efficiency and quality of the method in demanding situations. For these cases, we study convergence of the new scheme. We find it compatible with the discretization method used for the derivative operators. We employ new finite-difference operators to approximate second derivatives, and we see that using these operators, instead of, say, applying the first-order ones twice, we get a much more accurate approximation without any high frequency noise (which otherwise would be observed but could be removed by using Kreiss-Oliger dissipation of the appropriate order). We then discuss the accuracy of the approximation, i.e, its quality at a given finite (and reasonable) resolution.

In section 4 we study a much more demanding situation, specifically, an equation with variable coefficients both in space and time, which is often used in general relativity as a standard test, and show that this scheme is well behaved for this case too.

In all the cases we studied, we find that the new method compares favorably with the SAT scheme, producing a smaller error and using less information on the boundaries. The reason this happens, in part, is that second-order operators produce a significantly smaller phase error than first-order operators. These results show that our approach turns out to be a very competitive and easily implemented scheme.

2 Numerical scheme

We consider, for simplicity, a one dimensional problem, the generalization to more dimensions being trivial. Consider a field $\Phi(x,t) : S^1 \times \Re \to \Re$ satisfying the wave equation:

$$\partial_t^2 \Phi = \partial_x^2 \Phi, \tag{1}$$

and assume sufficiently smooth initial data is given at $t = 0$: $\Phi(x,0) = \Phi_0(x)$, $\partial_t \Phi(x,0) = \Pi_0(x)$.

As mentioned, the traditional way to solve this equation when interfaces are present is by reducing it to first-order form by introducing, for instance, the variables $\Pi := \partial_t \Phi$ and $\Psi := \partial_x \Phi$ (from here on we deal exclusively with interfaces of the type called *touching grids*, where for each boundary point of a grid

there corresponds another point from a neighboring grid representing the same spatial point). Then (1) is equivalent to the system

$$\begin{aligned} \partial_t \Phi &= \Pi, \\ \partial_t \Psi &= \partial_x \Pi, \\ \partial_t \Pi &= \partial_x \Psi. \end{aligned} \tag{2}$$

This way of solving the equation has the previously mentioned disadvantage of introducing auxiliary variables, something that can be very expensive in terms of memory, especially when considering systems of wave equations in many dimensions, as is often the case, for instance, in general relativity. Furthermore, the use of first-order systems results in less accurate numerical approximations, for the phase error is larger than when using schemes based on the second-order version of the systems.
We are interested here in solving (1) in second-order form for the spatial operators, although we shall keep the first-order form for the time integration, since we will use either a Runge-Kutta or IMEX scheme to advance the fields in time. We therefore consider the system

$$\begin{aligned} \partial_t \Phi &= \Pi, \\ \partial_t \Pi &= \partial_x^2 \Phi, \end{aligned} \tag{3}$$

$$\tag{4}$$

and develop a numerical method for solving it when interfaces are present.

Standard theorems guarantee the existence of a solution in the energy norm

$$\mathcal{E} := \int \{\Pi^2 + \nabla\Phi \cdot \nabla\Phi\} dV. \tag{5}$$

We want to develop a scheme that will preserve the analogous discrete-energy norm, thus guaranteeing stability.

In order to solve this system numerically, we take a uniformly spaced grid and set an interface at $x = 0$ that will connect one grid end with the other, resulting in a circle of length two. We write the discrete solution as a vector $\{\Phi_j\}$, $j = 0 \dots N$ corresponding to points $x_j = dx * j$ with $dx := \frac{2}{N}$, so the last point coincides with the first one.

We introduce the discrete l^2-norm in the usual fashion,

$$< \Psi, \Phi >:= dx \sum_{j=0}^{N} \sigma_j \Psi_j \Phi_j$$

where $\{\sigma_j\}$ are a set of real-valued weights that depend on the finite-difference operators under consideration and dx is the interspace between neighboring grid points.

The semi-discrete system we want to solve at all points, except at the boundary is then

$$\begin{aligned} \partial_t \Phi_j &= \Pi_j \\ \partial_t \Pi_j &= (D^2\Phi)_j \qquad j = 0 \dots N, \end{aligned} \tag{6}$$

where D is any finite-difference operator that approximates the derivative operator to some order $q \geq 1$ satisfying the summation-by-parts property (SBP from now on) [3, 4, 5]. That is, it satisfies the discrete counterpart of the integration-by-parts property

$$< \Psi, D\Phi > + < D\Psi, \Phi >:= \Psi_N \Phi_N - \Psi_0 \Phi_0.$$

Alternatively, we could use, instead of D^2, a second-order operator D_2 approximating the second derivative, which satisfies the corresponding SBP property, i.e. that can be written as [6]

$$D_2 = H^{-1} \left(-D^T H D + RS \right). \tag{7}$$

Here $H = diag(\sigma_0, ..., \sigma_N)$, D is an operator that approximates the first derivative, $R = diag(-1, 0, ..., 0, 1)$ and S approximates the first derivative at the boundary. This guarantees that the analogue of the integration-by-parts property for the second derivative holds, i.e, that D_2 satisfies

$$< \Psi, D_2\Phi >= \Phi_N(D\Psi)_N - \Phi_0(D\Psi)_0 - < D\Psi, D\Phi > . \tag{8}$$

This is a better choice, for the operators have a smaller stencil and preserve the solution phase more accurately. We shall use in this work the SBP second-order operators obtained in [7].

If we could prove that the linear ODE system (6) has eigenvalues with no positive real part and a complete set of eigenvectors, then there would be many discrete-time integrators, and as a result, a stable numerical evolution to the whole system. For a more detailed description of the theory see, for instance, [5]. A way to check those conditions is to find a norm that is constant or decreases in time. This is the procedure we shall use to implement our scheme. In particular we shall later use either traditional explicit Runge-Kutta third-order operators or new IMEX ones. These IMEX methods consist of a mixture of implicit and explicit Runge-Kutta methods. This allows us to solve the problem without having to invert the whole equation system (a property of the explicit schemes), while allowing at the same time for large negative eigenvalues (a property of implicit methods). The result is a very efficient penalty scheme.

It is clear then that if we use this scheme, we will get, for the discretized version of the energy norm,

$$\frac{\partial}{\partial t}\mathcal{E} := \Pi_N D\Phi_N - \Pi_0 D\Phi_0.$$

Since the contributions in the RHS of this equation come from each side of the boundary, in order to preserve this norm during evolution we need to cancel these boundary terms. In the traditional first-order implementation, this is achieved by means of the SAT method, which consists of adding penalty terms to the equation at the boundary points, causing the above energy to be preserved. In the SAT approach, [1], the first-order system (2) is semi-discretized and modified at the boundary as follows:

$$\begin{aligned}
\partial_t\Phi_j &= \Pi_j, \\
\partial_t\Psi_j &= D\Pi_j - \frac{1}{2}\frac{\delta_{j0}}{dx\,\sigma_0}\left((\Pi_0 - \Pi_N) + (\Psi_0 - \Psi_N)\right) \\
&\quad + \frac{1}{2}\frac{\delta_{jN}}{dx\,\sigma_N}\left((\Pi_N - \Pi_0) - (\Psi_N - \Psi_0)\right) \\
\partial_t\Pi_j &= D\Psi_j - \frac{1}{2}\frac{\delta_{j0}}{dx\,\sigma_0}\left((\Pi_0 - \Pi_N) + (\Psi_0 - \Psi_N)\right) \\
&\quad - \frac{1}{2}\frac{\delta_{jN}}{dx\,\sigma_N}\left((\Pi_N - \Pi_0) - (\Psi_N - \Psi_0)\right).
\end{aligned} \tag{9}$$

The result is a stable scheme with non-increasing energy. In contrast to this first-order hyperbolic and to the parabolic case, it does not seem possible for second-order systems to control the energy by introducing on each side terms proportional to the difference of the fields and their normal derivatives at each boundary. Thus, following [2] we introduce our first modification by adding terms at the boundary as follows:

$$\partial_t\Pi_j = (D^2\Phi)_j + \frac{1}{dx\,\sigma_0}\delta_{j0}(D\Phi)_0 - \frac{1}{dx\,\sigma_N}\delta_{jN}(D\Phi)_N.$$

With this modification the boundary terms cancel and so the total norm remains constant, but they introduce no interaction between the two sides of the interface, and so the solution we get would just bounce back at the boundary (the energy is conserved and if one boundary point can not possibly influence the point at the other side, the pulse has to bounce back). However, eliminating the boundary term means that we can now concentrate on adding terms that, while preserving/decreasing the norm, introduce an interaction at the

ends of the grid in such a way that the wave can pass through the interface. We must, therefore, introduce a term that couples the two sides, namely a penalty term that forces the values at both extremes to coincide. The simplest one that satisfies this property is

$$\partial_t \Pi_j = (D^2\Phi)_j + \frac{1}{dx\,\sigma_0}\delta_{j0}(D\Phi)_0 - \frac{1}{dx\,\sigma_N}\delta_{jN}(D\Phi)_N - L(\Pi_0 - \Pi_N)(\delta_{j0} - \delta_{jN}), \tag{10}$$

where L, which we call the *interaction factor*, is a positive real constant to be chosen as large as possible, in order to make the interaction as strong as possible. In this way we penalize the difference on the two sides of the interface and drive them to coincide with a very large exponential factor, while keeping the energy bounded. The limitation on how large L can be comes from the fact that a too-large value would make the system unstable by making a large contribution to the eigenvalues along the negative real axis, making explicit time integration schemes fall outside their stability region, or making the needed time step prohibitively small. For explicit schemes the value of L should not be larger than $L = \frac{1}{\sigma_0 dx}$, so it contributes to the CFL factor as much as the principal part. This turns out not to be good enough, giving unacceptably large errors in the form of bounces at the interface for a resolution that describes appropriately the solution. Thus we had to use larger factors and so to resort to a semi-implicit method which would free us from the CFL limitation.

The system of ordinary differential equations (6) described above, with the proposed correction (10), was evolved using Runge-Kutta type methods: first the usual third-order one, and then an IMEX method [8, 9], specifically the one called the IMEX-SSP3(4,3,3) L-stable scheme in [9]. For the spatial discretization we used finite-difference operators approximating the second derivative obtained in [7]. We also compared with the use of first derivative operators [10, 4, 11] applied twice. We did this because in some systems where off-diagonal terms occur in the Laplacian, or lower-order terms are present, one might want to use a single operator for every derivative. We then compare the results obtained with these methods with the evolution obtained using the standard first-order SAT method (9). We report on the findings in the next section.

3 Tests

We tested the method by running simulations both in $1D$ and $2D$. For the $1D$ case all the runs were performed on a circle of length 2 (i.e the domain was the interval $[0, 2]$, where the last grid point is identified with the first one). For the $2D$ simulations, on the other hand, the domain was a torus and the grid consisted of a 2×2 square with the $x = 0$ face identified with the $x = 2$ face, and similarly for the y coordinate. In this case, for simplicity, one of the interfaces, namely the one corresponding to the y direction, was treated using penalties, while for the x direction we used periodic operators. In all the runs the number of points and the order of the finite-difference operators employed guarantee a good enough resolution for cases where the solution has a high frequency[1].

3.1 Initial-data sets

For the purpose of analyzing convergence it is sufficient to choose **smooth** initial data. We therefore choose the following data

- **1D smooth initial data**

$$\Phi_0(x) := 4^{12}\, x^{12}(x-1)^{12},$$

$$\Pi_0(x) := \partial_x \Phi_0(x),$$

On the other hand, for the purpose of comparing realistic situations and in order to analyze how the method keeps the phase of the solution, we take the following **rough** and highly variable data:

[1] Here we aim at an accuracy of about one part in 10^3 for 10 periods. Enough to keep the phase without appreciable error for about 10 crossing times.

- **1D rough initial data**

$$\Phi_0(x) := e^{-8^2(x-0.5)^2}\cos(50\pi x),$$

$$\Pi_0(x) := \partial_x\Phi_0(x),$$

corresponding to a rough pulse propagating to the left.

- **2D rough initial data**

$$\Phi_0(x,y) := e^{-8^2((x-1.5)^2+(y-1.5)^2)}\cos(50\pi(y-x)),$$

$$\Pi_0(x,y) := \frac{1}{\sqrt{2}}(\partial_x\Phi_0(x,y) - \partial_y\Phi_0(x,y)),$$

3.2 Space discretizations

As mentioned, we performed runs for three different choices of the space discretization, as well as for the time integrator. For the space operators we compared the results to

- Traditional first-order reduction with boundaries treated using the SAT technique.
- Second-order formulation with a second-derivative operator D_2.
- Second-order formulation where the second derivative is approximated by the first-derivative operator D applied twice (i.e. D^2).

In all the runs we use a very accurate finite-difference operator, in particular, the first derivative operator is an optimized operator of order eight in the interior and order four at points in the boundary [10, 4, 11] (from now on we will call it the 8-4 operator). The second-derivative operator used is of order eight in the interior and order six at the boundary [7]- This operator comes also with a first-order companion which we used for the boundary contributions. Both of these, as well as the 8-4 ones, satisfy SBP. The choice of these operators was made in order to preserve the correct phase of the solution on long-time runs, and to be able to test the contribution to the error coming form the interaction term, with the smallest possible interference from the contribution to the error of the derivatives discretizations.

3.3 Time integration

As noted above, we used two time integrators, a traditional Runge-Kutta third-order scheme and an IMEX one. The necessity of an IMEX scheme comes from the fact that the interaction factor has to be very large, hence stiff, in order to achieve good accuracy. To show this, below we display a run of smooth data with an interaction factor of $L = \frac{1}{\sigma_0 dx}$. The runs of this section were performed with a resolution of 640 points and with $dt = 2.5 \times 10^{-5}$ (CFL = 0.008). In the plot below, Figure 1, we show both the periodic and the interface approximations. The extra bump to the right is the bounce of a fraction of the solution at the interface.

It is possible to reduce the error to a very small amount by enlarging the interaction factor, but at the expense of losing efficiency, since for the traditional Runge-Kutta scheme the time step needed for stability becomes significantly smaller. In fact we observed the errors to fall to very small values for an interaction factor a thousand times larger (with the drawback of having to use a time step a thousand times smaller). See Figure 2. To avoid small time steps, while allowing larger interaction factors, semi-implicit methods are needed. We shall use here a method among those called IMEX, [8, 9], specifically, the one called IMEX-SSP3(4,3,3) L-stable scheme in [9]. These methods permit us to explicitly solve stiff parts of the equations

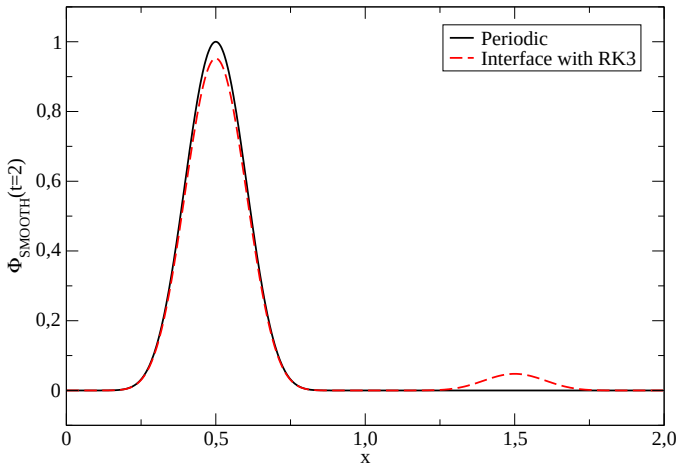

Figure 1: **Comparison of periodic and interface runs using the traditional 3^{th} order Runge-Kutta method. Here $L = 10 * dx^{-1}$.**

while keeping the other terms as usual in traditional Runge-Kutta schemes. So, from now on, for all runs with the new interface method (for the second-order version of the wave equation) we present the results obtained with the IMEX time integrator, while for all runs for the SAT method (i.e., for first-order version of the wave equation) and all the periodic runs were performed with the traditional third-order Runge-Kutta integrator. Note that, the only term that needs to be treated implicitly with this IMEX method is the term proportional to L, that is, just the last boundary term in (10).

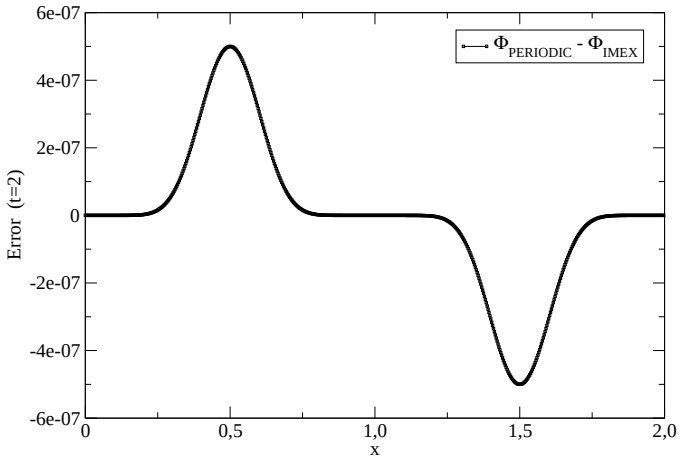

Figure 2: **Error of the interface run (i.e, difference with the periodic run), using a Runge-Kutta time integrator with an interaction factor $L = 10^6 * dx^{-1}$.**

3.4 Convergence

In the absence of the interaction term, we expect the error to be of the form $e = f_1 dt^p + f_2 dx^q$, where p depends on the time integrator used and q on the space discretization of derivatives.

The convergence rate is calculated as

$$Q = ln\left(\frac{\|\Phi^{(h)} - \Phi^{(h/2)}\|_{l_2}}{\|\Phi^{(h/2)} - \Phi^{(h/4)}\|_{l_2}}\right)/ln(2), \tag{11}$$

where $\Phi^{(h_i)}$ is the numerical solution with grid spacing h_i.

In our case, we expect $p \geq 3$ for the IMEX algorithm. The precise value depends on the nature of the solution, in particular the size of the solution near the boundary (where the implicit part of the algorithm is used) in comparison with the size of the solution in the interior of the grid. Furthermore, $q \geq 5$ since the derivatives used are fourth-order accurate at the boundary and eighth-order accurate in the interior. For stability reasons, the CFL condition on the explicit integrator is such that we need to scale dt as dx, so we expect a convergence index of the order of three. Alternatively, we might fix a sufficiently small dt and increase the space resolution, which allows us to study in an independent way space convergence. In this case we would expect a convergence index of the order of five. Any smaller convergence factor must result from the interface treatment.

For most of the convergence tests we used very smooth initial data, since the f_1 and f_2 functions depend on high derivatives of the exact solution. We start by analyzing the convergence of the method for the $1D$ case with runs of 640, 1280 and 2560 points using the smooth data. From Figure 3, we see that, for the first-order treatment and the D^2 second-order formulation keeping CFL constant (0.08), the convergence factor starts at a value of 3 while the pulse is in the interior, meaning that the main contribution to the error comes from time discretization. By the time the solution reaches the boundary the Q factor climbs to ~ 5, which means that there the space discretization is the primary contribution to the error. For the D_2 second-order case, however, the convergence remains constant around 3, implying that during the whole run the derivative operator's contribution to the error is negligible.

On the other hand, for fixed $dt = 2.5 \times 10^{-5}$, we observe that during the whole run the error is dominated by the space operators, and the convergence factor starts at a high value, close to 8, corresponding to the time when the pulse has not reached the boundary; and falling to 5 when it does.

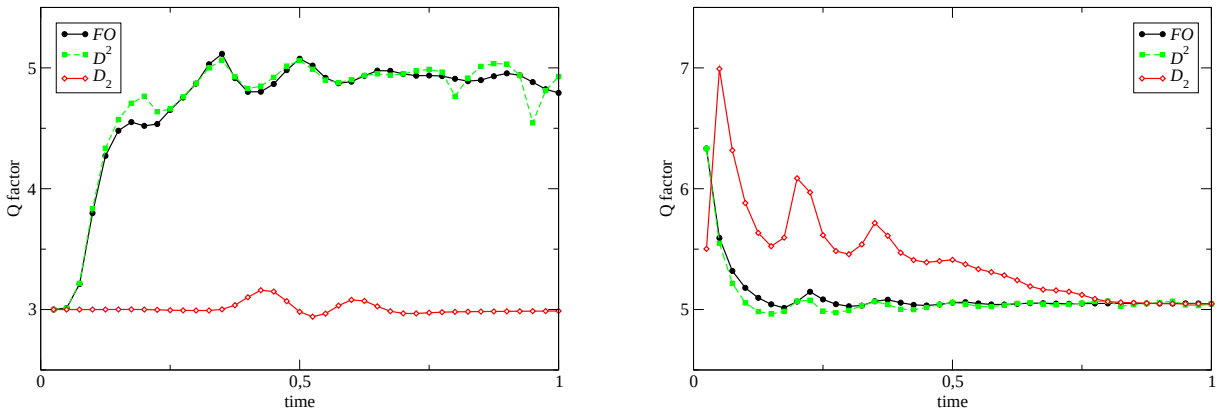

Figure 3: **Comparative plot of the convergence factor for the $1D$ system, for: first-order formulation FO, second-order formulation using D^2 and second-order formulation using D_2, at (a) CFL, and (b) dt fixes.**

For the $2D$ case we performed runs of 640×640, 1280×1280 and 2560×2560 points with the rough data.

From Figure 4, we observe very similar behavior as in the $1D$ case. Before the wave reaches the boundary, for all discretizations using the third-order Runge-Kutta integrators, both conventional or IMEX, convergence is dominated by the time discretization with a Q factor close to 3, climbing to ~ 5 as the pulse reaches the interface (where the space discretization is the one contributing the most to the error). We also performed a run, for comparison, using a fourth-order Runge-Kutta method for the first-order system. In this case we observe that in the interior the convergence improves and starts close to 8. Here the time integrator is more accurate and hence the space discretization becomes more important. As the pulse reaches the boundary, we again obtain a Q factor of 5.

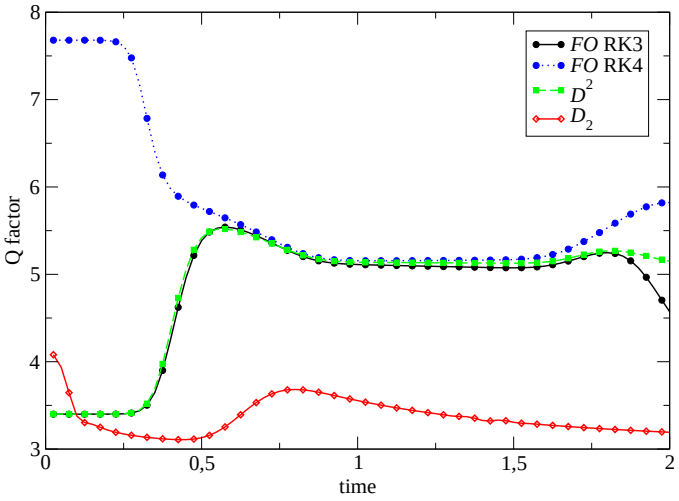

Figure 4: **Comparative plot of the convergence factor for the $2D$ system, at fixed CFL $= 0.08$. Plotted are the first-order formulation FO, second-order formulation using D^2 and second-order formulation using D_2.**

Convergence alone is not enough to guarantee that we are approaching the correct solution. This is because our system is not a priori consistent. That is, in principle, the limit of our finite-difference scheme does not need to coincide with the continuum equation (because of the boundary terms which grow with resolution). Thus, it is necessary to analyze convergence against the true solution, which we do in the next section.

3.5 Accuracy

Here we compare methods for realistic data, namely the rough initial data given above, both for the $1D$ and $2D$ cases.

For the $1D$ case we evolved the solution up to $t = 2.0$, at which point the solution has moved to the left and the pulse has completely passed the interface located at $x = 0$. In the $2D$ case a pulse is sent in an oblique direction to the interface to check whether the scheme preserves the correct phase in this case and does not introduce, for instance, an excess bounce.

For comparison we performed a run using periodic boundary conditions with 8^{th} order centered-difference operators with $N = 5120$ points, or 5120×5120 for the $2D$ case (referred to as P_{5120} in both cases). This was used as the reference solution against which we compare all the other runs. For these last simulations, interface conditions were used with $N = 640$, 1280 and 2560 points for both the first-order system and the second-order one (denoted by FO_{640}, FO_{1280}, FO_{2560} and $D_{2\,640}$, $D_{2\,1280}$, $D_{2\,2560}$ respectively). In addition,

for the second-order system, we performed simulations using both D^2, and D_2 operators to approximate the second derivative in the RHS. All the runs were performed with an interaction factor of $10^6 * dx^{-1}$ and keeping the CFL factor constant (0.08).

In Figure 5, we show a comparison of the l^2-norm of the error for the two different cases under consideration: the standard first-order system with a third-order Runge-Kutta time integrator, and our second-order system with the D_2 operator that uses the IMEX-SSP3(4,3,3) L-stable scheme. Note that before the pulse has reached the interface, the two methods are comparable, but as soon as the wave reaches and passes through the boundary, the solution obtained using our second-order method improves the accuracy by at least one order of magnitude. This shows that the interface treatment proposed here competes very well with the traditional SAT scheme.

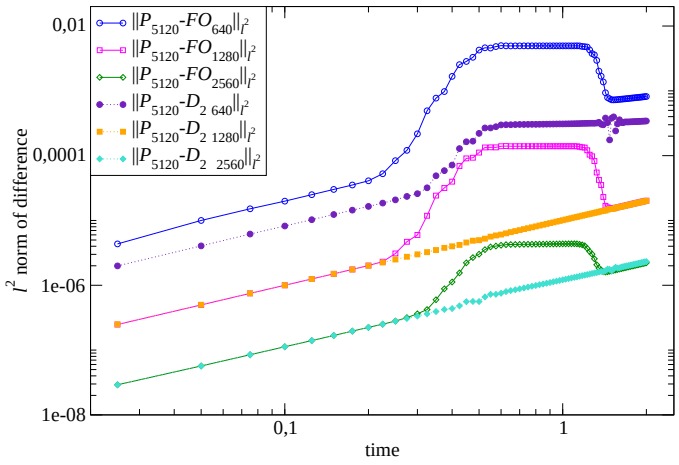

Figure 5: **l^2-norm of the error for several different $1D$ runs.**

We observe the same behavior for the $2D$ case, displayed in Figure 6. Again our method and the standard SAT method behave similarly in the interior region, but ours is superior to the SAT after the pulse passes the interface.

3.6 Energy decay

The present scheme is energy-diminishing at the semi-discrete approximation level. This implies that if a stable time integrator is used with a sufficiently small time step the energy given by (5) should decrease only at a rate given by the penalty term, plus, perhaps noticeable, the inherent dissipation of the time integrator. So here we study such a decay, showing that it is indeed very small, as one would infer from the method's accuracy.

Figure 7, shows the behavior of the relative error of the energy on longer runs: ten times the previous ones. These runs were performed with fixed CFL $= 0.08$, and a resolution of 5120 points. As expected, the decay is very small, and it improves considerably for larger values of the interaction factor. For a value of $L = 10^6 * dx^{-1}$, the energy decays at a faster rate than with the first-order SAT scheme, which coincides with the decay given by a periodic treatment. However, if we increase the value of L, the decay

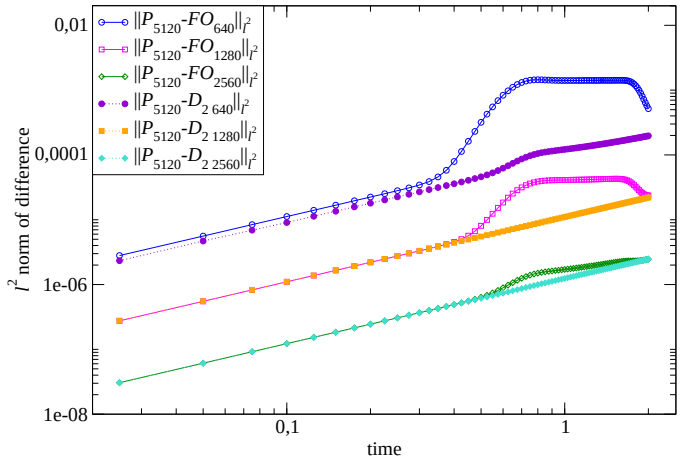

Figure 6: **l^2-norm of the error for several different $2D$ runs.**

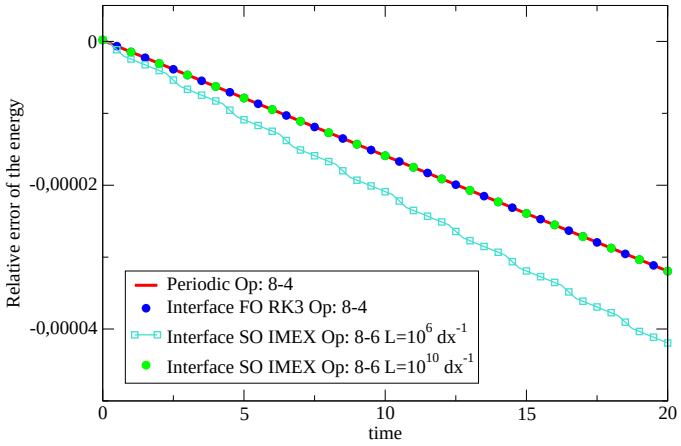

Figure 7: **Relative error of the energy decay for the rough initial data for four different scenarios: periodic boundary conditions, traditional SAT first-order method, and our second-order scheme using D_2 for two choices of the interaction factor L.**

approaches the periodic one, and if we take $L = 10^{10} * dx^{-1}$ the three decays (periodic, FO-SAT, and SO) are indistinguishable. So most of the decay is due to the inherent-Runge-Kutta integrators, and both the standard third-order and the IMEX one seem to have the same dissipation.

Finally, in Figure 8 we show the relative error of the energy compared to that of the periodic solution, i.e. $(E - E_{periodic})/E_{periodic}$, in order to account only for the decay associated to the method. Here we see that the first-order method is the one that best approximates the periodic energy, while our second-order method deviates from it. This difference decreases, however, if we take a larger L, showing, once more, that the larger the interaction factor, the better the proposed method fits the solution.

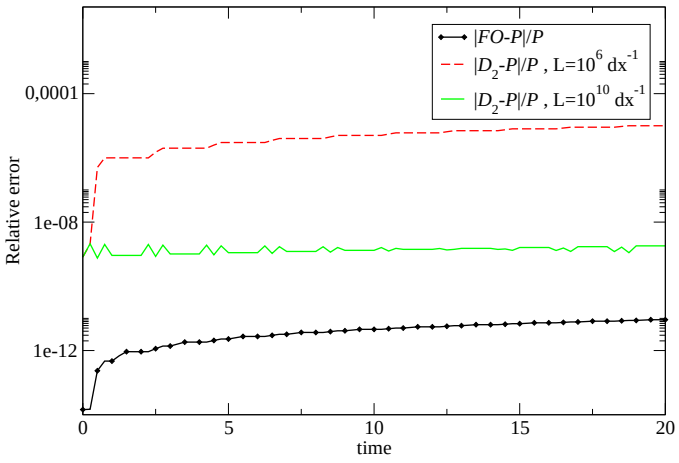

Figure 8: **Energy relative error (compared to the periodic-solution energy) for the rough initial data for three different scenarios: traditional SAT first-order method, and our second-order scheme using D_2 for two choices of the interaction factor L.**

3.7 Dissipation

It is worth noticing that, for all the runs performed so far, it was not necessary to introduce any artificial dissipation, for we have been considering a linear problem with constant coefficients, and therefore there was no noise introduced by high frequency modes.

However, for other choices of spatial operators, or if we were dealing with a nonlinear equation or one with non-constant coefficients, we might find high frequency oscillations around the correct solution. As an example of this, we used the D^2 scheme instead of the D_2 used above, with the rough data. We see in Figure 9 that using this operator introduces some numerical noise to the solution, diminishing its quality. We note that the solutions without dissipation are almost indistinguishable except near the interface, where we included a zoomed sector to show the disagreement. We see that this noise is removed by using Kreiss-Oliger dissipation [5], that is, by adding to the equations a term proportional to a large power of the Laplacian operator. This term contains a factor that depends on the resolution in such a way as to make the error produced by adding this term to be of the same or smaller order as that of the rest of the terms in the approximation. In particular, we used the one that corresponds to the accuracy we are using for the finite-difference operators [12, 10, 11], namely, eighth-derivative dissipation $Q_d = -\sigma dx^9 \Delta^4$, where Δ is a finite-difference operator that approximates the Laplacian to first-order accuracy. The runs used for this comparison were performed with a resolution of 640 points at CFL fix (0.08), and $\sigma = 100$.

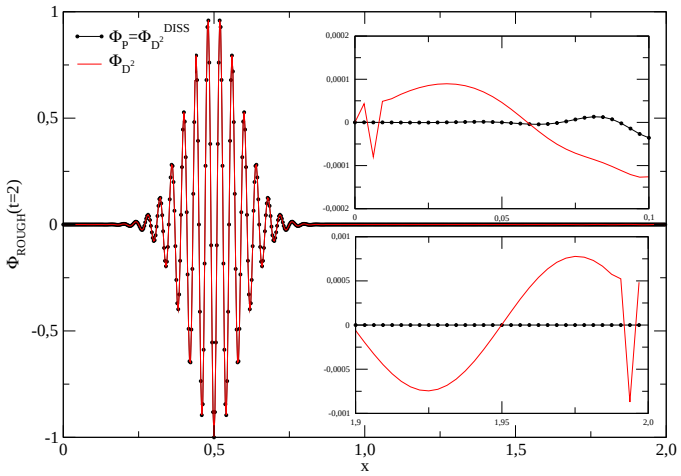

Figure 9: **Comparison of the solution at $t = 2$ using D^2 for the spatial operator with and without dissipation, with the periodic one.**

Figure 10 shows the l^2-norm of the error using D^2 with and without dissipation, as well as the error for the FO system with dissipation. We calculated the error by comparing a periodic run with 5120 points against interface runs with 640, 1280 and 2560 points. These were done keeping the CFL factor fix (0.08) and using a interaction factor of $10^6 * dx^{-1}$. We see that adding the dissipative term improves the accuracy by one or two orders of magnitude. The errors calculated with dissipation for both FO and SO systems are almost the same. Also, by comparing with Figure 6, we see that the errors for the D^2 and FO methods with a dissipation are similar to the error calculated with the second-order operator D_2. Thus, methods that use dissipation are competitive with the D_2 discretization.

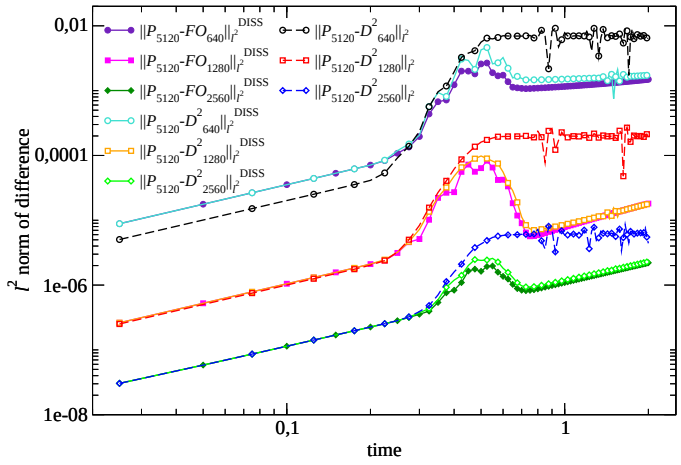

Figure 10: **Behavior of the relative error of the l^2-norm for runs with operator D^2 with and without dissipation.**

4 Applications

As an application of our method involving variable coefficients, we consider a simple $1D$ test in numerical relativity: a linearized solution of Einstein's equations around a gauge-wave background with line element [13]

$$ds^2 = e^{A\,sin(\pi(x-t))}(-dt^2 + dx^2) + dy^2 + dz^2. \tag{12}$$

This background describes flat spacetime, in which a coordinate transformation on the (t, x) plane has been performed, with a sinusoidal dependence along $t - x$. This gauge-wave problem provides us with a simple, yet non trivial, numerical test, for it is linear, the amplitude of the coefficients can be controlled by only adjusting the parameter A, and does not lead to any singularities. This test differs from those of the previous sections since in this case the coefficients depend both on space and time.

There are various papers that deal with this problem [10, 13, 14, 15]. Most of them use a method that involves a first-order formulation with periodic boundary conditions, except for [10], which uses a boundary treatment. A second-order scheme with boundary conditions for this gauge-wave problem was studied in [16, 17]. One aspect that these papers show is the exponential growth and loss of convergence displayed by the solution for large amplitudes.

In this section we will apply the method developed above to analyze this problem. We use the same approach as [10] where perturbations of (12) are considered and introduced in Einstein's equations in order to derive the linearized evolution equations for the fields. Here we study the short-time behavior since we are only interested in the stability of the method as the waves go through the interface.

The non trivial variables for this problem are the relevant components of the metric and its time derivative (g_{xx}, K_{xx}), and the lapse α. We consider, therefore, perturbations of the form

$$g_{xx} = e^{A\,sin(\pi(x-t))} + \delta g_{xx} \tag{13}$$

$$K_{xx} = \frac{A}{2}\,cos(\pi(x-t))e^{\frac{A}{2}\,cos(\pi(x-t))} + \delta K_{xx} \tag{14}$$

$$\alpha = e^{\frac{A}{2}\,sin(\pi(x-t))} + \delta\alpha \tag{15}$$

The resulting equations are

$$\partial_t\Phi = \Pi + A\pi\,cos(\pi(x-t)),$$

$$\partial_t\Pi = \frac{1}{\hat{\alpha}}\partial_x(\hat{\alpha}\partial_x\Phi) - \frac{1}{2}\left(A\pi^2\,sin(\pi(x-t)) + \frac{A^2\pi^2}{2}cos^2(\pi(x-t))\right)\Phi \tag{16}$$

$$-\frac{1}{2}A\pi\,cos(\pi(x-t))\Pi, \tag{17}$$

where $\hat{\alpha} = e^{\frac{A}{2}sin(\pi(x-t))}$, $\Phi = \delta\alpha/\hat{\alpha}$ and $\Pi = \delta K_{xx}$.

We performed several runs and compared the results for the two second-order formalisms treated in this paper, namely, the first-derivative operator applied twice (D^2) and the second-derivative operator (D_2). The semidiscretization of the second-derivative term in (16) is thus of the form

$$\partial_x(\hat{\alpha}\partial_x\Phi) \sim D(\hat{\alpha}D\Phi) \tag{18}$$

for the D^2 case. For the D_2 case, on the other hand, we split the second derivative as

$$\partial_x(\hat{\alpha}\partial_x\Phi) \sim \hat{\alpha}D_2\Phi + (\partial_x\hat{\alpha})D_1\Phi \tag{19}$$

where the derivative of $\hat{\alpha}$ is calculated analytically, and D_1 and D_2 are fully compatible finite-difference operators approximating the first and second derivative respectively [7].

In figures (11) and (12) we show the convergence factor and the l^2-norm of the error for these two cases. The runs were performed in the $[-1, 1]$ interval, using an amplitude of 0.5, CFL = 0.01 and the following smooth initial data

$$\Phi_0(x) := 100^{12}\,(x+0.6)^{12}(x+0.4)^{12}, \tag{20}$$
$$\Pi_0(x) := \partial_x \Phi_0(x). \tag{21}$$

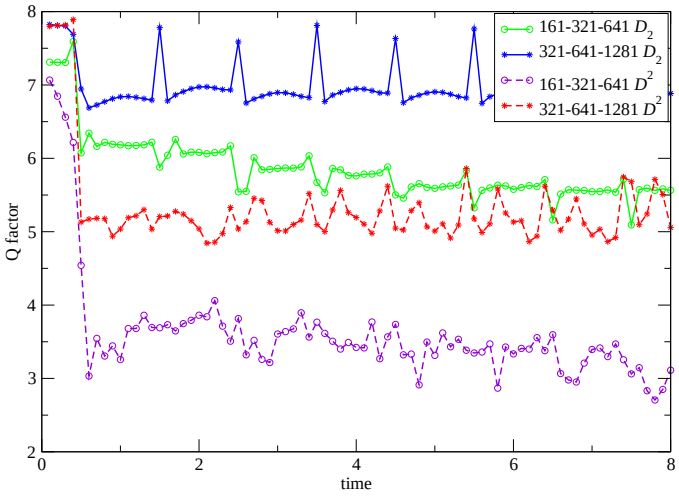

Figure 11: **Convergence factor for two different resolutions of the D^2 and D_2 formalisms.**

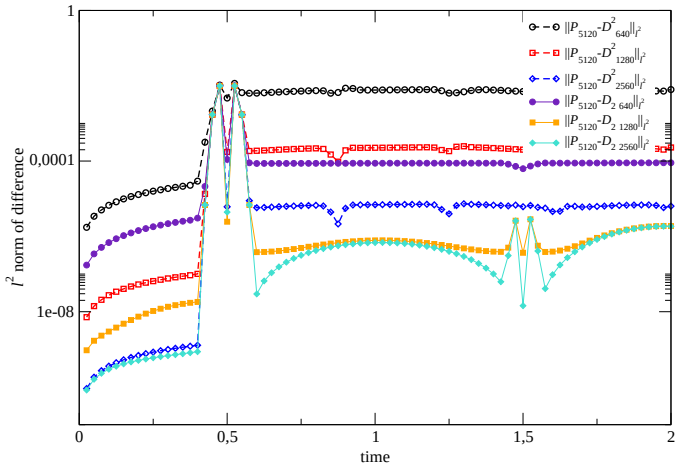

Figure 12: **l^2-norm of the error for various resolutions for the two second-order formulations.**

For all the runs, except for the D^2 case with the lowest resolution, the convergence factors oscillate between the expected values of 8 and 5, consistent with the order of the difference operators used. The l^2-norm of the error shows again that the D_2 formulation has a lower error compared to the D^2 case for all

the resolutions considered. This illustrates the superior performance of the present method even for variable coefficients, which opens a wide range of possible applications.

5 Conclusions

We have shown that it is possible to implement an interface scheme of the "penalty" type for the second-order wave equation similar to the ones used for first-order hyperbolic and parabolic equations. This scheme shares with them similar properties: Only data at points at the interface need to be passed between grids, and convergence is ensured for linear, constant-coefficient systems. This scheme was applied as well to a problem with non-constant coefficients, namely, perturbations of a gauge-wave background. The accuracy of the method seems to be as good as the accuracy of the finite-difference operators and of the time integrators used, and competes favorably with the usual first-order SAT schemes for all the cases we have tried.

Note that for the wave equation in n dimensions one would have to pass at the boundary $n+1$ quantities, namely the space and time derivatives of the fields, while in our case one only needs to pass the values of the time derivative. This is important for multi-block parallelizations for it implies that one obtains the same quality for a solution sharing only a very small fraction of the data one would need for a comparable (in accuracy) SAT scheme. This will considerably improve the scale properties of multi-block MPI computations. It might even be advantageous to use it when dividing a grid block into many smaller grids to be dealt by different MPI processes. In this case the traditional way of doing it is to pass at the boundary the whole stencil needed to compute finite differences using centered operators. The accuracy of our method implies one could just pass among the neighboring grids the values of the fields, gaining a substantial step on scalability.

Since the information passed along the interface is a time derivative, it behaves as a scalar with respect to coordinate changes in space [2]. So, its values at both sides of the grid, namely at two different coordinate patches, should be identified without any change. By contrast, when using traditional SAT schemes and passing space derivatives of the fields, a coordinate transformation is needed in the generic case at which the boundary regions represent different coordinate patches. Thus the new scheme requires less coding and less computation. Furthermore, in non-trivial situations the SAT method requires the computation of the incoming modes at the boundary points to impose the correct penalties, in cases where the wave-equation systems are quasilinear, such as, for instance, general relativity, this is a major time-consuming task which can be also spared with the present scheme.

This new method is not unique to second-order systems, for its underlying ideas can be applied to many cases of interest. In particular, it can be extended to the general case of symmetric hyperbolic first-order system. This case is under present investigation.

6 Acknowledgments

We thank Luis Lehner for discussions, and SeCyT-UNC, CONICET, FONCyT and the Partner Group grant of the Max Planck Institute for Gravitational Physics (Albert Einstein Institute) for financial support. O.R. thanks Perimeter Institute for hospitality, where part of this research was carried out.

References

[1] Mark H. Carpenter, Jan Nordström, and David Gottlieb. A Stable and Conservative Interface Treatment of Arbitrary Spatial Accuracy. *Journal of Computational Physics*, 148(2):341–365, 1999.

[2] O. Reula. Numerical treatment of interfaces in Quantum Mechanics. *arXiv*, quant-ph/1103.5448:1–11, 2011.

[2] This is of course true in the case of scalar quantities. In the case we were dealing with systems of wave equations applied to tensor quantities, some coordinate transformations are unavoidable at interfaces.

[3] H.-O. Kreiss and G. Scherer. On the Existence of Energy Estimates for Difference Approximations for Hyperbolic Systems. *Tech. Rep. Dept. of Scientific Computing, Uppsala University*, 1977.

[4] Bo Strand. Summation by Parts for Finite Difference Approximations for d/dx. *Journal of Computational Physics*, 110(1):47–67, 1994.

[5] B. Gustafsson, H.-O. Kreiss, and J. Oliger. *Time Dependent Problems and Difference Methods*. Wiley, New York, 1995.

[6] K. Mattsson and J. Nordström. Summation by parts operators for finite difference approximations of second derivatives. *J. Comput. Phys.*, 199:503–540, 2004.

[7] Ken Mattsson and Florencia Parisi. Stable and Accurate Second-Order Formulation of the Shifted Wave Equation. *Commun. Comput. Phys.*, 7:103–137, 2010.

[8] Uri M. Ascher, Steven J. Ruuth, and Raymond J. Spiteri. Implicit-Explicit Runge-Kutta methods for time-dependent partial differential equations. *Applied Numerical Mathematics*, 25(2-3):151–167, 1997.

[9] Lorenzo Pareschi and Giovanni Russo. Implicit-Explicit Runge-Kutta Schemes and Applications to Hyperbolic Systems with Relaxation. *Journal of Scientific Computing*, 25:129–155, 2005.

[10] Luis Lehner, Oscar Reula, and Manuel Tiglio. Multi-block simulations in general relativity: high order discretizations, numerical stability, and applications. *Class. Quant. Grav.*, 22:5283–5322, 2005.

[11] P. Diener, E. N. Dorband, E. Schnetter, and M. Tiglio. Optimized high-order derivative and dissipation operators satisfying summation by parts, and applications in three-dimensional multi-block evolutions. *J. Sci. Comput.*, 32:109–145, 2007.

[12] Kreiss H. and Oliger J. Methods for the Approximate Solution of Time Dependent Problems. *GARP Publication Series*, 10, 1973.

[13] M. Tiglio, L. Lehner, and D. Neilsen. 3D simulations of Einstein's equations: symmetric hyperbolicity, live gauges and dynamic control of the constraints. *Phys.Rev. D*, 70, 2004.

[14] G. Calabrese, J. Pullin, O. Sarbach, and M. Tiglio. Stability properties of a formulation of Einstein's equations. *Phys.Rev. D*, 66, 2002.

[15] Alcubierre M. *et al*. Toward standard testbeds for numerical relativity. *Class.Quant.Grav.*, 21, 2004.

[16] B. Szilágyi, H-O. Kreiss, and J. Winicour. Modeling the Black Hole Excision Problem. *Phys.Rev. D*, 71, 2005.

[17] M. C. Babiuc, B. Szilágyi, and J. Winicour. Harmonic Initial-Boundary Evolution in General Relativity. *Phys.Rev. D*, 73, 2006.